Benefit of data clustering
for osm2pgsql/mapnik rending

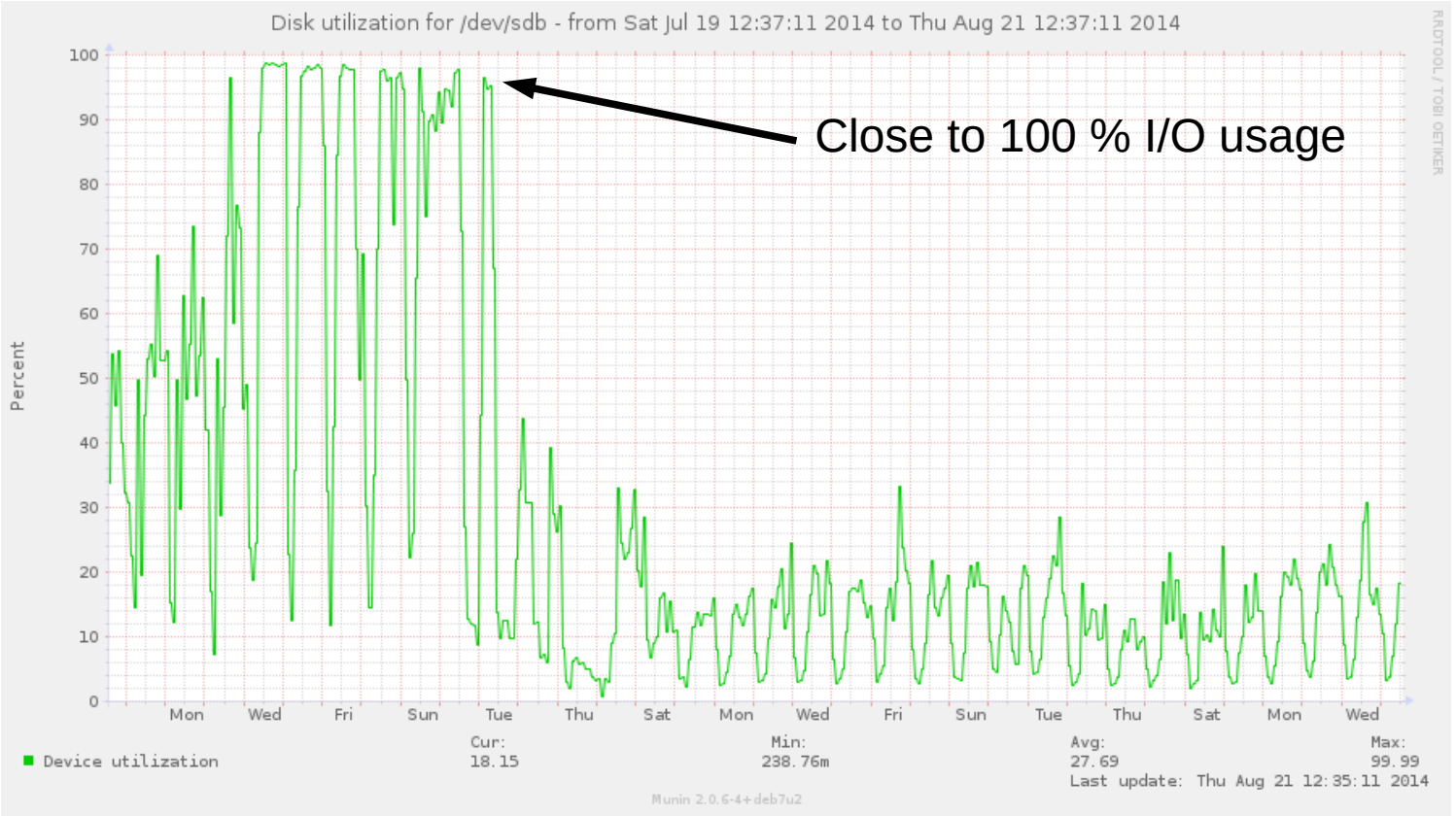
Christian Quest - @cq94
cquest@openstreetmap.fr

Database fragmentation

- Similar problem to « disk fragmentation »
- Data spread all over disk sectors
- Dramatically increases I/O count

OSM-FR main tile server

- July 2014 : close to 100 % I/O bandwidth used
  - SSD could not keep up with tile rendering request



How does Postgres store data ?

- 8 KB at a time… these are postgres « pages »
- A page is filled with data until it is full
- No default mechanism to gather data on the same page

How does mapnik uses data ?

- Objects located in a given bbox are needed to render one (meta) tile
- These objets are usually store in many different postgres 8KB pages
- Many pages to access in order to get all objects

Let's optimize this !

- Make sure nearby objects are stored on the 8KB pages
- Fewer pages will have to be read from disk to access the same objets

Postgres CLUSTER !

- Postgres CLUSTER command allows to reorder data based on one index
- CLUSTER makes a copy of the data using the index
  - 1st step : CREATE INDEX
  - 2nd step : CLUSTER

Which index ?

- PostGIS geometric index is not optimal
  - Creates « rectangle » based indexes
  - Our tiles are more « square » than « rectangles »

Geohash to the rescue !

- Geohash are a text version of close to square bounding boxes
- Longer geohash = smaller boxes
- Postgis inclused ST_Geohash function !

Check geohash on wikipedia for more details...

Which geohash length ?

- Zoom 18 tiles = 2^18
- Zoom 18 metatiles = 2^15
- Each geohash char adds 2.5 bits
- 15 / 2.5 = 6 → 6 chars are enough !
- Helps limit index size (has hash computation)

Optimizing index creation

ST_Geohash needs WGS84 input data…

For nodes (easy) :

- ST_Geohash(ST_Transform(node,4326),6)

```
CREATE INDEX planet_osm_point_cluster ON
planet_osm_point
(ST_Geohash(ST_Transform(way,4326),6));
```

Optimizing index creation

For ways (and polygons) :

- ST_Geohash(ST_Transform(way,4326),6)

  → ST_Transform computes each node WGS84 location !!!

- ST_Expand(way,0) → get the way bbox, then compute Geohash

```
CREATE INDEX planet_osm_line_cluster ON
planet_osm_line
(ST_Geohash(ST_Transform(ST_Expand(way,0),4326),6));
```

Let's CLUSTER !

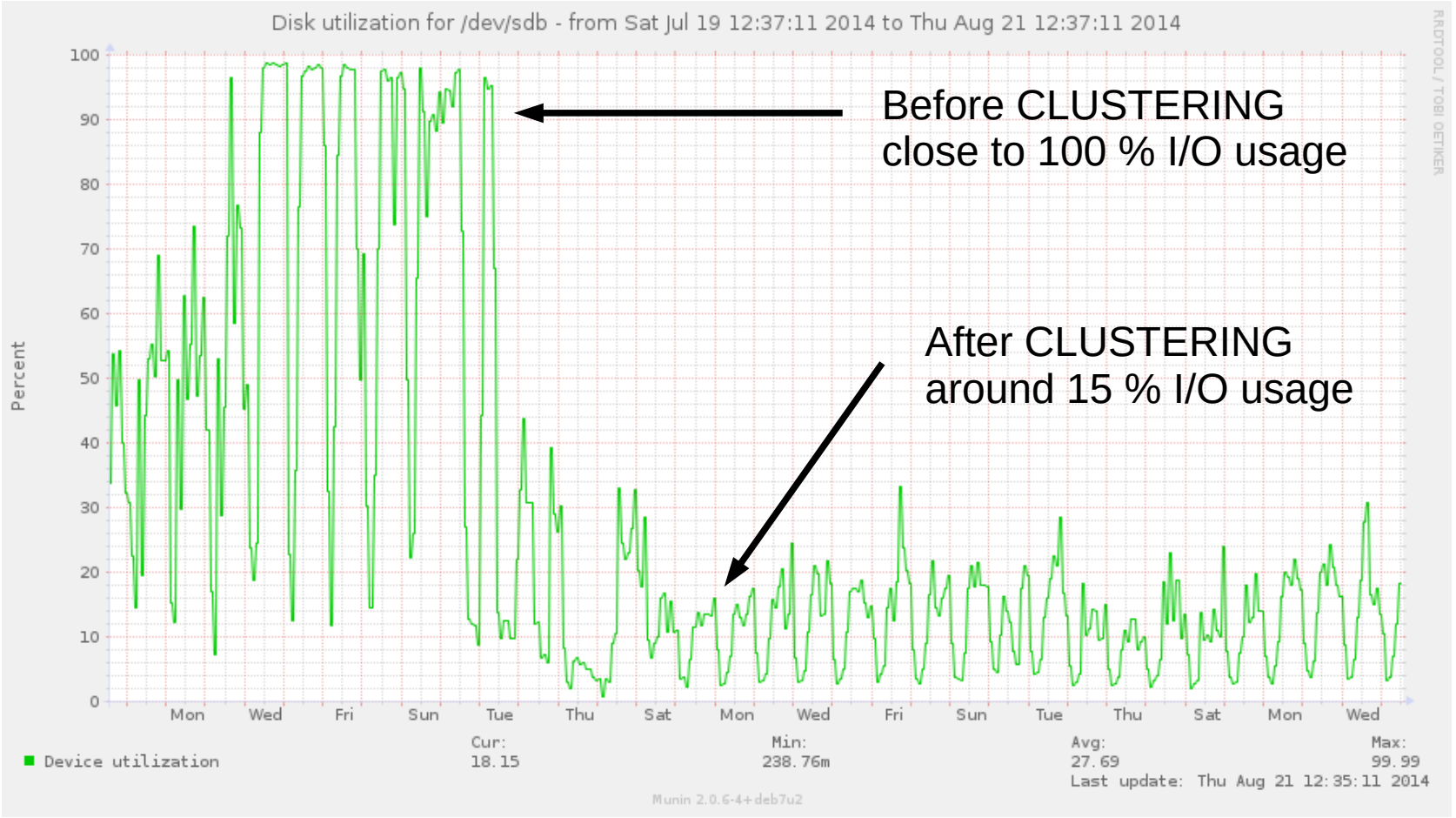
Traps :

- locks may occur during the CLUSTER (suspend diff updates!)
- CLUSTER is making a full copy of the data and needs disk space for that
- it may take a long time... several hours for the planet_osm_polygon table in my case

```
CLUSTER planet_osm_point USING planet_osm_point_cluster;
CLUSTER planet_osm_roads USING planet_osm_roads_cluster;
CLUSTER planet_osm_line USING planet_osm_line_cluster;
CLUSTER planet_osm_polygon USING planet_osm_polygon_cluster;
```

I/O reduced ?
100 % down to 15 % !

Check the stats again…



(Positive) Side effects...

- Useful data density increase in data pages
- Less data pages to read from disk to render a given metatile
- Disk cache can hold more useful data in RAM

  → Even less disk I/O !

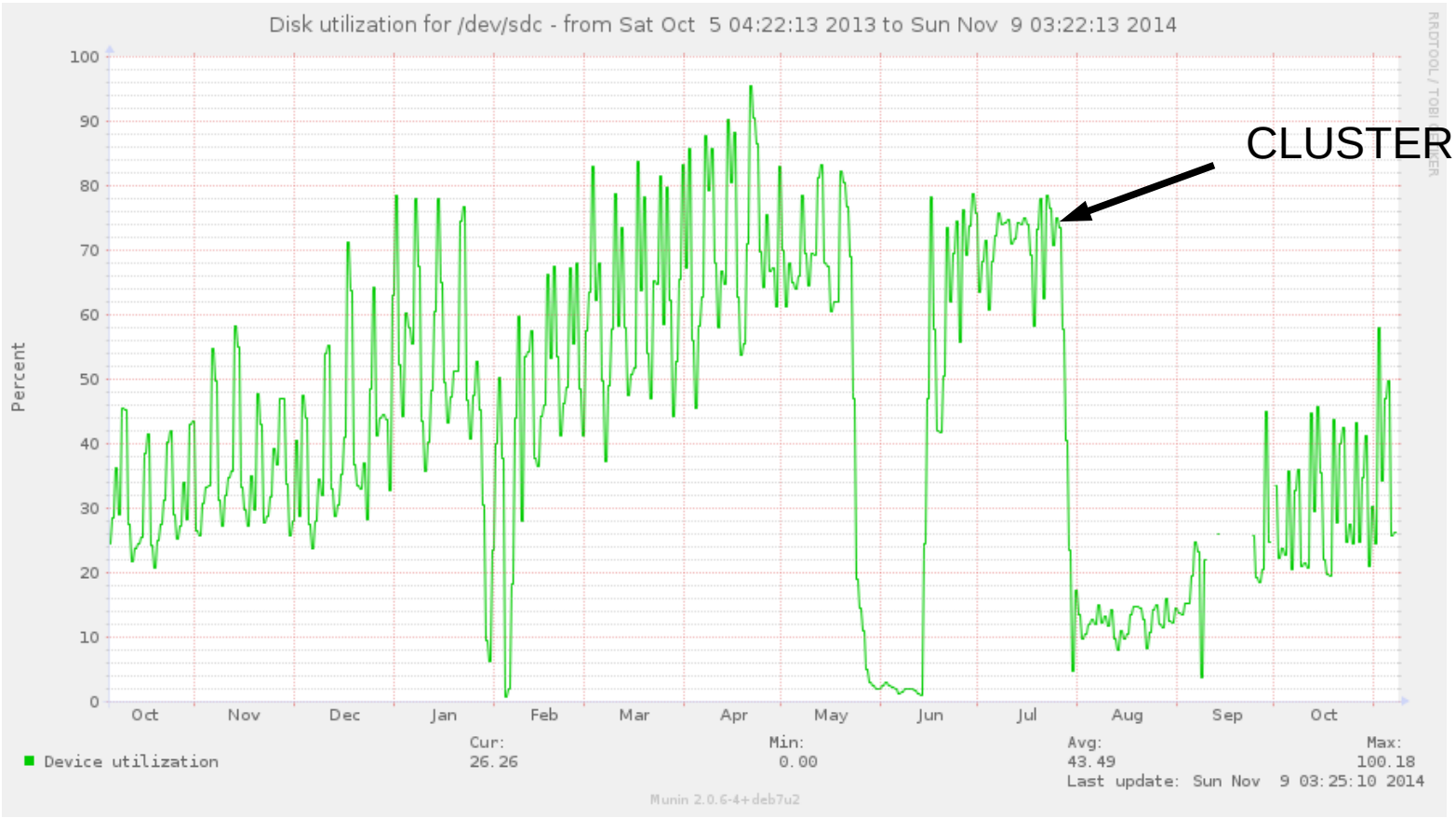
On the long term...

- CLUSTER reorganize data on disk only at the time it is used
- New and updated data will not respect the CLUSTER ordering

  → Need to re-CLUSTER from time to time...

Time to re-CLUSTER ?

- I/O increasing after a few months of update...



Questions ?

Christian Quest

OSM : cquest / twitter : @cq94

Email : cquest@openstreetmap.fr